1 Polynomial Identity Testing

An equation on \( n \) variables is \textit{valid} if it holds true on all assignments of the variables. Testing if an equation is valid, is the polynomial identity testing problem.

\begin{center}
\begin{tabular}{|c|}
\hline
\textbf{Problem: Polynomial identity testing (PIT)}  \\
Input: polynomial \( p(x_1, \ldots, x_n) \)  \\
Question: Is \( p \equiv 0 \) (or conversely, do there exist \( x_1, \ldots, x_n \) so that \( p(x_1, \ldots, x_n) \neq 0 \))
\hline
\end{tabular}
\end{center}

We assume the variables, coefficients and exponents of \( p \) are all integers.

The input of PIT is given as an \textit{algebraic circuit}. An algebraic circuit is similar to a boolean circuit. Instead of performing boolean operations, the gates can do arithmetic operations. It consists of

- input variables \( x_1, \ldots, x_n \)
- constants \( 1, 0, -1 \)
- gates \( g_1, \ldots, g_m \). Each gate is defined by one of two equations
  1. \( g_i = g_j + g_k \)
  2. \( g_i = g_j \cdot g_k \)
- output gate \( g_m = p_m(x_1, \ldots, x_m) \)

By its definition, an algebraic circuit computes a polynomial on variables \( x_1, \ldots, x_n \).

\underline{Schwartz-Zippel-DeMillo-Lipton algorithm}

Let \( M \leftarrow 4n2^m \), and \( \ell \leftarrow 3m \).
Repeat \( T \) times:

- Pick random variables \( a_1, \ldots, a_n \in \{0, \ldots, M\} \).
- Pick a random prime \( Q \in \{2^\ell, \ldots, 2^{\ell+1} - 1\} \).
- If \( p(a_1, \ldots, a_n) \mod Q = 0 \), then continue.
Else reject.
Output “probably valid”.

Degree of the polynomial computed by the circuit:

• Input variables: degree = 1.
• Constants: degree = 0.
• Sum gates: \( \deg(g_i) = \max(\deg(g_j), \deg(g_k)) \)
• Product gates: \( \deg(g_i) = \deg(g_j), \deg(g_k) \leq 2 \max(\deg(g_j), \deg(g_k)) \).

Overall, \( \deg(p(x_1, \ldots, x_n)) \leq 2^m \).

**Lemma 1.** If each \( a_i \) is selected uniformly at random from \( \{0, \ldots, M\} \), then

\[
\Pr[p(a_1, \ldots, a_n) = 0] \leq \frac{1}{4}.
\]

To prove Lemma 1, define \( p_i(x_{i+1}, \ldots, x_n) = p(a_1, \ldots, a_i, x_{i+1}, \ldots, x_n) \).

**Lemma 2.** \( \forall i \),

\[
\Pr[p_i \neq 0 \text{ but } p_{i+1} \equiv 0] \leq \frac{1}{4n}.
\]

**Proof of Lemma 2**

\[
p_i(x_{i+1}, \ldots, x_n) = \sum_{\vec{e} = (e_{i+1}, \ldots, e_n)} c(e_{i+1}, \ldots, e_n) \prod_{j=i+1}^{n} x_j^{e_j} = \sum_{\vec{e} = (e_{i+2}, \ldots, e_n)} \prod_{j=i+2}^{n} x_j^{e_j} (q_{\vec{e}}(x_{i+1})) \tag{1}
\]

Each polynomial on \( x_{i+1} \) has \( \deg(q_{\vec{e}}) \leq 2^m \), so it has at most \( 2^m \) roots.

If \( p_i \neq 0 \),

\[
\Pr[p_{i+1} \equiv 0] \leq \Pr[q_{\vec{e}}(a_{i+1}) = 0] \leq \frac{\deg(q_{\vec{e}})}{M} \leq \frac{2^m}{4n \cdot 2^m} = \frac{1}{4n}.
\]

Lemma 1 is implied by Lemma 2 by the union bound.

Let \( A = p(a_1, \ldots, a_n) \). An error occurs when \( A \neq 0 \), but \( A \mod Q = 0 \), i.e. \( Q | A \). We factor \( A \) so that \( A = q_1^{f_1} \cdots q_k^{f_k} \), where \( q_1, \ldots, q_k \) are prime factors of \( A \). An obvious lower bound is \( A \geq 2^k \), or \( k \leq \log A \). An error occurs only when \( Q \in \{q_1, \ldots, q_k\} \). So we need to argue that \( k \) is not large.

The maximum value produced by the arithmetic circuit is \( M^{2^m} \) (when all inputs are \( M \) and all gates are multiplication). So \( A \leq M^{2^m} \), and then \( k \leq \log A \leq 2^m \log M \leq 2^m \).

---

1Explanation of equation (1):

Left: \( e_j \) is the exponent of \( x_j \). A sum over all \( (e_{i+1}, \ldots, e_n) \) is the sum of all monomials on variables \( x_{i+1}, \ldots, x_n \).

\( c(e_{i+1}, \ldots, e_n) \) is the coefficient of the corresponding polynomial.

Right: for all monomials on variables \( x_{i+1}, \ldots, x_n \), we gather the terms, and thus get a univariate polynomial on \( x_{i+1} \) for each monomial on \( x_{i+2}, \ldots, x_n \).

Example: \( x_1^2 x_2 x_3 + x_1 x_2 x_3 + x_1 x_3 + x_1 x_3 + x_2 = x_2 x_3 (x_1^2 + x_1) + x_2 (x_1 + 1) + x_3 (x_1) \).
Theorem 3 (Prime Number Theorem). The number of $\ell$-bit primes is $\Theta(2^\ell / \ell)$.

We set $\ell = 3m$. Then the total number of $\ell$-bit primes is $2^\ell / \ell$, which is much greater than $2^{2m}$, the number of false positives that are prime.

Lemma 4. If $A \neq 0$, $\text{Prob}[A \mod Q \neq 0] \geq \Omega(1/\ell)$.

Proof. By Theorem 3, if we pick a random $Q$, $\text{Prob}[Q \text{ is prime}] = \Theta(1/\ell)$.

The conditional probability

$$\text{Prob}[A \mod Q = 0 | Q \text{ is prime}] \leq \frac{\# \text{ prime factors of } A}{\# \text{ primes } \in [2^\ell, 2^{\ell+1} - 1]}$$

Thus $\text{Prob}[A \mod Q \neq 0] \geq \Omega(1/\ell)$.

The probability of finding a counter-example if invalid: $\Omega(1/m)$.

If $p(x_1, \ldots, x_n) \neq 0$, each time, the probability of rejection $\geq \Omega(1/m)$. After $T$ loops, the probability of never rejecting is at most $(1 - c/m)^T \leq e^{-cT/m} = e^{-m}$, for $T = m^2/c$.

2 Probabilistic complexity classes

For a randomized algorithm $A(x, r)$, where $x$ is the actual input and $r$ is the random choices, define $A(x) = \text{Prob}_r[A(x, r)] \in [0, 1]$.

Definition 5. $L \in \text{BPP}$ (bounded-error probabilistic poly-time) if there exists a polynomial time algorithm $A$ so that
- $\forall x \in L, A(x) > 3/4$.
- $\forall x \notin L, A(x) < 1/4$.

Definition 6. $L \in \text{RP}$ if there exists a polynomial time algorithm $A$ so that
- $\forall x \in L, A(x) > 3/4$.
- $\forall x \notin L, A(x) = 0$.

$L \in \text{co-RP}$ if there exists a polynomial time algorithm $A$ so that
- $\forall x \in L, A(x) = 1$.
- $\forall x \notin L, A(x) < 1/4$.

Example 2.1. PIT $\in \text{co-RP}$.

Lemma 7. $\text{RP} \subseteq \text{NP}$.

Definition 8. $L \in \text{ZPP}$ if there exists a polynomial time algorithm $A$ so that
- $\forall x \in L$, it returns either “yes” or “don’t know”. The probability of “yes” is at least $1/2$.
- $\forall x \notin L$, it returns either “no” or “don’t know”. The probability of “no” is at least $1/2$. 
